IN THE CLAIMS

1. (original) A method in a data processing system for developing a data flow program comprising code segments, the method comprising the steps of:

dividing into blocks a memory area that extends over a data set comprising data operated on by the data flow program;

for each block in the memory area, associating data from the data set with the block and associating at least one code segment with the block;

storing data read and data write identifiers for each code segment, the data read and data write identifiers identifying at least a portion of the data set read or written by the code segment;

determining dependencies between blocks based on the read and write identifiers;

displaying a directed acyclic graph, the directed acyclic graph comprising nodes and arcs, each node representing at least one block, and each arc representing a determined dependency;

initiating execution of the code segments; and

while the code segments are executing,

determining whether a debugging command has been received; and

when it is determined that a debugging command has been received,

processing the debugging command.

2. (currently amended) A method in a data processing system for developing a data flow program comprising code segments that operate on data in memory, the method comprising the steps of:

dividing a memory area into blocks and associating each block with at least a portion of the data and with at least one code segment;

storing data read and data write identifiers for each code segment, the data read and data write identifiers identifying at least a portion of the data read or written by the code segment;

determining dependencies between blocks based on the read and write identifiers;

generating a graph representation of the data flow program, the graph representation comprising nodes associated with the blocks, and dependencies between the blocks that provide an execution order for the code segments;

executing a debugging command on the data flow program.

3. (original) A method according to claim 2, wherein the debugging command comprises a replay debugging command.

4. (original) A method according to claim 2, wherein the debugging command comprises a dependency modification debugging command.

5. (original) A method according to claim 2, wherein the debugging command comprises a step debugging command.

6. (original) A method according to claim 2, wherein the debugging command comprises a breakpoint debugging command.

7. (original) A method according to claim 2, wherein the debugging command comprises a single point breakpoint command.

8. (original) A method according to claim 2, wherein the debugging command comprises a none after breakpoint command.

9. (original) A method according to claim 2, wherein the debugging command comprises an all before breakpoint command.

10. (original) A method according to claim 2, wherein the debugging command comprises a task node breakpoint command.

11. (original) The method of claim 2, wherein the debugging command comprises a combination single point breakpoint and none after breakpoint command.

12. (original) The method of claim 2, wherein the debugging command comprises a combination single point breakpoint and all before breakpoint command.

13. (original) The method of claim 2, wherein the debugging command comprises a combination single point breakpoint, none after breakpoint, and all before breakpoint command.

14. (original) The method of claim 2, further comprising the step of displaying the graph representation and emphasizing the breakpoint node using a breakpoint visualization.

15. (original) The method of claim 2, wherein the debugging command comprises a dependency modification debugging command, and further comprising the step of:

reordering the nodes for execution to reflect a dependency change.

16. (original) The method of claim 2, wherein the debugging command comprises a step debugging command specifying a selected node associated with a selected node code segment and further comprising the steps of:

determining dependencies for the selected node;

executing nodes to satisfy the dependencies; and

executing the selected node code segment.

17. (original) The method of claim 2, wherein the debugging command comprises a replay debugging command and further comprising the step of:

saving graph status information in secondary storage.

18. (original) The method of claim 2, wherein the debugging command comprises a replay debugging command and further comprising the steps of:

retrieving graph status information from secondary storage; and

replaying execution of the nodes in accordance with the graph status information.

19. (original) The method of claim 18, wherein the graph status information includes node timestamps of execution and wherein the step of replaying comprises the step of graphically emphasizing the nodes in the graph representation based on the node timestamps of execution.

20. (currently amended) A method in a data processing system for developing a data flow program comprising nodes, the method comprising the steps of:

initiating execution of the data flow program; and

executing a debugging command on the data flow program,

wherein dependencies between nodes are determined based on data read and data write identifiers for code segments associated with the respective nodes, the data read and data write identifiers identifying at least a portion of data read or written by the code segment.

21. (original) A method according to claim 20, wherein the step of initiating execution comprises the steps of determining the data dependencies between the nodes and enqueueing the nodes in accordance with the data dependencies.

22. (original) A method according to claim 20, wherein the step of executing a debugging command comprises the step of executing a breakpoint debugging command.

23. (original) A method according to claim 20, wherein the step of executing a debugging command comprises the step of executing a dependency modification debugging command.

24. (original) A method according to claim 20, wherein the step of executing a debugging command comprises the step of executing a step debugging command.

25. (original) A method according to claim 20, wherein the step of executing a debugging command comprises the step of executing a replay debugging command.

26. (currently amended) A computer-readable medium containing instructions that cause a data processing system to perform a method for developing a data flow program comprising code segments that operate on data in memory, the method comprising the steps of:

dividing a memory area into blocks and associating each block with at least a portion of the data and with at least one code segment;

storing data read and data write identifiers for each code segment, the data read and data write identifiers identifying at least a portion of the data read or written by the code segment;

determining dependencies between blocks based on the data read and data write identifiers;

generating a graph representation of the data flow program, the representation comprising nodes associated with the blocks, and dependencies between the blocks that provide an execution order for the code segments;

executing a debugging command on the data flow program.

27. (original) A computer-readable medium according to claim 26, wherein the debugging command comprises a replay debugging command.

28. (original) A computer-readable medium according to claim 26, wherein the debugging command comprises a dependency modification debugging command.

29. (original) A computer-readable medium according to claim 26, wherein the debugging command comprises a step debugging command.

30. (original) A computer-readable medium according to claim 26, wherein the debugging command comprises a breakpoint debugging command.

31. (original) The computer-readable medium of claim 26, wherein the breakpoint debugging command comprises a single point breakpoint command.

32. (original) The computer-readable medium of claim 26, wherein the breakpoint debugging command comprises a none after breakpoint command.

33. (original) The computer-readable medium of claim 26, wherein the breakpoint debugging command comprises a all before breakpoint command.

34. (original) The computer-readable medium of claim 26, wherein the breakpoint debugging command comprises a task node breakpoint command.

35. (original) The computer-readable medium of claim 26, wherein the breakpoint debugging command comprises a combination single point breakpoint and none after breakpoint command.

36. (original) The computer-readable medium of claim 26, wherein the breakpoint debugging command comprises a combination single point breakpoint and all before breakpoint command.

37. (original) The computer-readable medium of claim 26, wherein the breakpoint

debugging command comprises a combination single point breakpoint, none after breakpoint, and all before breakpoint command.

38. (original) The computer-readable medium of claim 26, wherein the debugging command comprises a dependency modification debugging command, and wherein the method further comprises the step of:

reordering the nodes for execution to reflect a dependency change.

39. (original) The computer-readable medium of claim 26, wherein the debugging command comprises a step debugging command specifying a selected node associated with a selected node code segment and wherein the method further comprises the steps of:

determining dependencies for the selected node;

executing nodes to satisfy the dependencies; and

executing the selected node code segment.

40. (original) The computer-readable medium of claim 26, wherein the debugging command comprises a replay debugging command and wherein the method further comprises the step of:

saving graph status information in secondary storage.

41. (original) The computer-readable medium of claim 26, wherein the debugging command comprises a replay debugging command and wherein the method further comprises the steps of:

retrieving graph status information from secondary storage; and

replaying execution of the nodes in accordance with the graph status information.

42. (original) The computer-readable medium of claim 41, wherein the graph status information includes node timestamps of execution and wherein the step of replaying comprises the step of graphically emphasizing the nodes in the graph as the nodes executed based on the node timestamps of execution.

43. (currently amended) A data processing system comprising:

a memory comprising a data flow program and a data flow development tool that

associates data processed by the data flow program with blocks in the memory, that associates code segments of the data flow program to the blocks, that stores data read and data write identifiers for each code segment, the data read and data write identifiers identifying at least a portion of the data read or written by the code segment; that determines dependencies between the blocks that provide an execution order for the blocks based on the data read and data write identifiers, that executes code segments in parallel using multiple threads, and that executes debugging commands on the data flow program; and

a processor that runs the data flow development tool.

44. (original) The data processing system of claim 43, wherein the debugging command comprises a step debugging command.

45. (original) The data processing system of claim 43, wherein the debugging command comprises a replay debugging command.

46. (original) The data processing system of claim 43, wherein the debugging command comprises a single point breakpoint command.

47. (original) The data processing system of claim 43, wherein the debugging command comprises a none after breakpoint command.

48. (original) The data processing system of claim 43, wherein the debugging command comprises an all before breakpoint command.

49. (original) The data processing system of claim 43, wherein the debugging command comprises a task node breakpoint command.

50. (original) The data processing system of claim 43, wherein the debugging command is a dependency modification debugging command.

51. (original) The data processing system of claim 43, wherein the debugging command comprises a step debugging command.

52. (original) The data processing system of claim 43, wherein the debugging command comprises a replay debugging command.

53. (currently amended) A data processing system for developing a data flow program, the data processing system comprising:

means for apportioning a memory area into regions and associating data and code segments of a data flow program with the regions;

means for storing data read and data write identifiers for each code segment, the data read and data write identifiers identifying at least a portion of the data read or written by the code segment;

means for determining dependencies between the regions based on the data read and data write identifiers; and

means for executing debugging commands on the data flow program.

54. (currently amended) A computer readable memory device encoded with a data structure accessed by a data flow development tool run by a processor in a system, the data structure comprising:

nodes assigned to at least a portion of data processed by a data flow program comprising code segments, the nodes also assigned to at least one code segment;

data read and data write identifiers for each code segment, the data read and data write identifiers identifying at least a portion of the data read or written by the code segment;

dependencies between nodes determined based on the data read and data write identifiers; and

debugging information specified by a debugging command, wherein the data flow development tool accesses the data structure to execute the debugging command on the data flow program.

55. (original) The computer readable memory device of claim 54, wherein the debugging information comprises an identification of a breakpoint node.

56. (original) The computer readable memory device of claim 54, wherein the debugging command is a step debugging command and wherein the debugging information comprises an identification of a selected node specified by the step debugging command.